Realizability Problem for Constraint LTL

Ashwin Bhaskar, M. Praveen, CMI

TIME 2022 Symposium

Constraints and Constraint Systems

- A constraint system is a tuple $(D, <, =)$ where D is an infinite set and $<$ and $=$ are interpreted as linear order and equality relations over D.

Constraints and Constraint Systems

- A constraint system is a tuple $(D, <, =)$ where D is an infinite set and $<$ and $=$ are interpreted as linear order and equality relations over D.

- A term t is an expression of the form $X^i x$ where x is a variable.

Constraints and Constraint Systems

- A constraint system is a tuple $(D, <, =)$ where D is an infinite set and $<$ and $=$ are interpreted as linear order and equality relations over D.

- A term t is an expression of the form $X^i x$ where x is a variable.

- A constraint c is a relation either of the form $t_1 < t_2$ or $t_1 = t_2$.

Syntax of Constraint LTL

- Constraint linear-time temporal logic (CLTL) is an extension of LTL where Boolean propositions are replaced with such constraints.

Syntax of Constraint LTL

- Constraint linear-time temporal logic (CLTL) is an extension of LTL where Boolean propositions are replaced with such constraints.

- A CLTL formula ϕ is defined as:

$$\phi ::= c \mid \neg\phi \mid \phi \vee \phi \mid X\phi \mid \phi U \phi$$

  where c is any constraint.

Syntax of Constraint LTL

- Constraint linear-time temporal logic (CLTL) is an extension of LTL where Boolean propositions are replaced with such constraints.
- A CLTL formula ϕ is defined as:

$$\phi ::= c \mid \neg\phi \mid \phi \vee \phi \mid X\phi \mid \phi U \phi$$

  where c is any constraint.
- $F\phi$ and $G\phi$ - derived operators as used in LTL.

Semantics

- CLTL formulas are interpreted over sequences of valuations of the variables over the domain D.

Semantics

- CLTL formulas are interpreted over sequences of valuations of the variables over the domain D.

- For example: $\phi = G(x < Xy)$

Semantics

- CLTL formulas are interpreted over sequences of valuations of the variables over the domain D.

- For example: $\phi = G(x < Xy)$

$$\begin{array}{cccccccc} x & 1 & 2 & 3 & 4 & 5 & \ldots & \\ & & & & & & & \models \phi \\ y & 2 & 4 & 6 & 8 & 10 & \ldots & \end{array}$$

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1
y_e	3

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1
y_e	3
x_s	2
y_s	6

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1	2
y_e	3	4
x_s	2	
y_s	6	

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1	2
y_e	3	4
x_s	2	4
y_s	6	8

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1	2	3
y_e	3	4	5
x_s	2	4	
y_s	6	8	

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1	2	3
y_e	3	4	5
x_s	2	4	6
y_s	6	8	10

CLTL Game

- Given a CLTL formula ϕ, we define a 2-player game.
- Environment player owns variables x_e and y_e. System player owns x_s and y_s.
- For example: $\phi = G((x_e < x_s) \wedge (y_e < Xy_s))$

x_e	1	2	3	$\cdots$
y_e	3	4	5	$\cdots$
x_s	2	4	6	$\cdots$
y_s	6	8	10	$\cdots$

Realizability Problem

- Given a CLTL formula ϕ, the system is said to win a play of the CLTL game if the sequence of valuations satisfies ϕ.

Realizability Problem

- Given a CLTL formula ϕ, the system is said to win a play of the CLTL game if the sequence of valuations satisfies ϕ.

- The system is said to have a winning strategy if it is possible for the system to win every play of the game regardless of how the environment plays.

Realizability Problem

- Given a CLTL formula ϕ, the system is said to win a play of the CLTL game if the sequence of valuations satisfies ϕ.
- The system is said to have a winning strategy if it is possible for the system to win every play of the game regardless of how the environment plays.
- Given a CLTL formula ϕ and an ownership of the variables, the realizability problem refers to the problem of checking whether the system has a winning strategy in the CLTL game.

Single-sided CLTL Games

- In a single-sided CLTL game, the set of variables is split into two types, lookahead and future-blind.

Single-sided CLTL Games

- In a single-sided CLTL game, the set of variables is split into two types, lookahead and future-blind.

- The CLTL formula is such that the constraints in the formula cannot compare the values of the future-blind variables across different positions.

Single-sided CLTL Games

- In a single-sided CLTL game, the set of variables is split into two types, lookahead and future-blind.
- The CLTL formula is such that the constraints in the formula cannot compare the values of the future-blind variables across different positions.
- In the single-sided game, the environment is constrained to only own future-blind variables. The system is free to own both lookahead and future-blind variables.

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1
y_e	3

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1
y_e	3
x_s	4
y_s	6

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1	2
y_e	3	4
x_s	4	
y_s	6	

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1	2
y_e	3	4
x_s	4	8
y_s	6	8

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1	2	3
y_e	3	4	5
x_s	4	8	
y_s	6	8	

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1	2	3
y_e	3	4	5
x_s	4	8	12
y_s	6	8	10

Single-sided CLTL games

- Consider $\phi_1 = G((x_e < x_s) \wedge (y_e < Xy_s))$. Here the current value of y_e can be compared with the value of y_s at the next position. So y_e is a lookahead variable and the CLTL game is not single-sided.
- Now, let x_e, y_e, x_s be future-blind variables and y_s be a lookahead variable. Consider the single-sided CLTL game with winning condition $\phi_2 = G((x_e < x_s) \wedge (y_e < x_s) \wedge (y_s < Xy_s))$.

x_e	1	2	3	$\cdots$
y_e	3	4	5	$\cdots$
x_s	4	8	12	$\cdots$
y_s	6	8	10	$\cdots$

Our Results

We prove that the realizability problem for CLTL is:

Our Results

We prove that the realizability problem for CLTL is:

- Undecidable for integers with linear order and equality

Our Results

We prove that the realizability problem for CLTL is:

- Undecidable for integers with linear order and equality
- 2EXPTIME-complete for single-sided games on integers with linear order and equality

Undecidability over $(\mathbb{Z}, <, =)$

- We prove undecidability of the realizability problem over $(\mathbb{Z}, <, =)$ using a standard game-theoretic technique where we reduce the reachability problem for 2-counter machines to the realizability problem.

Undecidability over $(\mathbb{Z}, <, =)$

- We prove undecidability of the realizability problem over $(\mathbb{Z}, <, =)$ using a standard game-theoretic technique where we reduce the reachability problem for 2-counter machines to the realizability problem.

- Both players participate in the simulation of the 2-counter machine. The counter value c_i is simulated as $x_i - y_i$ where x_i is an environment variable and y_i is a system variable for $i \in \{1, 2\}$.

Undecidability over $(\mathbb{Z}, <, =)$

- We prove undecidability of the realizability problem over $(\mathbb{Z}, <, =)$ using a standard game-theoretic technique where we reduce the reachability problem for 2-counter machines to the realizability problem.

- Both players participate in the simulation of the 2-counter machine. The counter value c_i is simulated as $x_i - y_i$ where x_i is an environment variable and y_i is a system variable for $i \in \{1, 2\}$.

- CLTL formulas are used to control how the variable values are assigned so that the transitions of the counter machine are simulated faithfully.

Decidability for single-sided games

- The realizability problem for CLTL games is undecidable in the general case, but we gain decidability for the single-sided case.

Decidability for single-sided games

- The realizability problem for CLTL games is undecidable in the general case, but we gain decidability for the single-sided case.

- In order to prove decidability, we use the technique of abstracting the concrete models using symbolic models.

Symbolic Models

- Given a CLTL formula ϕ, any sequence of valuations of the variables that satisfies the formula is called a concrete model. It is a sequence over an infinite domain as the number of valuations is infinite.

Symbolic Models

- Given a CLTL formula ϕ, any sequence of valuations of the variables that satisfies the formula is called a concrete model. It is a sequence over an infinite domain as the number of valuations is infinite.
- For example: Given $\phi = G((x > Xx) \wedge (y < Xy) \wedge (x < y))$, we have the following concrete model satisfying it:

Symbolic Models

- Given a CLTL formula ϕ, any sequence of valuations of the variables that satisfies the formula is called a concrete model. It is a sequence over an infinite domain as the number of valuations is infinite.

- For example: Given $\phi = G((x > Xx) \wedge (y < Xy) \wedge (x < y))$, we have the following concrete model satisfying it:

$$\begin{array}{cccccc} x & -1 & -2 & -3 & -4 & \dots \\ y & 1 & 2 & 3 & 4 & \dots \end{array} \models \phi$$

Symbolic Models

- A symbolic model abstracts the concrete model to a model over a finite domain.

Symbolic Models

- A symbolic model abstracts the concrete model to a model over a finite domain.
- The finite domain consists of all those constraints whose lengths do not exceed the length of the largest constraint in ϕ.

Symbolic Models

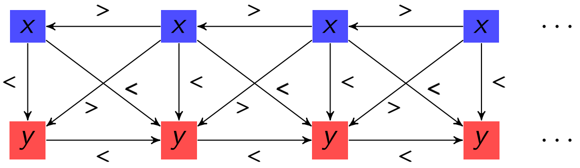
- A symbolic model abstracts the concrete model to a model over a finite domain.
- The finite domain consists of all those constraints whose lengths do not exceed the length of the largest constraint in ϕ.
- The symbolic model is a sequence of the set of such constraints satisfied at each position.

Symbolic Models

- A symbolic model abstracts the concrete model to a model over a finite domain.
- The finite domain consists of all those constraints whose lengths do not exceed the length of the largest constraint in ϕ.
- The symbolic model is a sequence of the set of such constraints satisfied at each position.
- For eg., we have the following symbolic model for the formula ϕ that abstracts any concrete model that satisfies ϕ:

Symbolic Models

- A symbolic model abstracts the concrete model to a model over a finite domain.
- The finite domain consists of all those constraints whose lengths do not exceed the length of the largest constraint in ϕ.
- The symbolic model is a sequence of the set of such constraints satisfied at each position.
- For eg., we have the following symbolic model for the formula ϕ that abstracts any concrete model that satisfies ϕ:



Symbolic Model Technique

- This technique has already been used to prove that the satisfiability problem for this logic is decidable.[1]

[1] Stéphane Demri and Deepak D'Souza. An automata-theoretic approach to constraint ltl. Information and Computation, 205(3):380–415, 2007

Symbolic Model Technique

- This technique has already been used to prove that the satisfiability problem for this logic is decidable.[1]
- The main technical contribution of our work is that we lift this symbolic model technique to CLTL games.

[1] Stéphane Demri and Deepak D'Souza. An automata-theoretic approach to constraint ltl. Information and Computation, 205(3):380–415, 2007

Symbolic Model Technique

- This technique has already been used to prove that the satisfiability problem for this logic is decidable.[1]
- The main technical contribution of our work is that we lift this symbolic model technique to CLTL games.
- We can think of strategies in the CLTL game as an infinitely branching tree with labels from an infinite alphabet.

[1] Stéphane Demri and Deepak D'Souza. An automata-theoretic approach to constraint ltl. Information and Computation, 205(3):380–415, 2007

Symbolic Model Technique

- This technique has already been used to prove that the satisfiability problem for this logic is decidable.[1]
- The main technical contribution of our work is that we lift this symbolic model technique to CLTL games.
- We can think of strategies in the CLTL game as an infinitely branching tree with labels from an infinite alphabet.
- We show that using the symbolic model technique, it is possible to reason using finitely branching trees with labels from a finite alphabet.

[1] Stéphane Demri and Deepak D'Souza. An automata-theoretic approach to constraint ltl. Information and Computation, 205(3):380–415, 2007

Future Work

- Single-sided CLTL realizability problem is 2EXPTIME-complete. We would like to check if there are expressive fragments of CLTL with lower complexity that work on practical examples.

Future Work

- Single-sided CLTL realizability problem is 2EXPTIME-complete. We would like to check if there are expressive fragments of CLTL with lower complexity that work on practical examples.

- We believe that single-sided CLTL games over the natural numbers is also decidable. We plan to prove it by appropriately extending the techniques that we have used to prove decidability over the integers.

Thank You!